FaS 7차 세미나

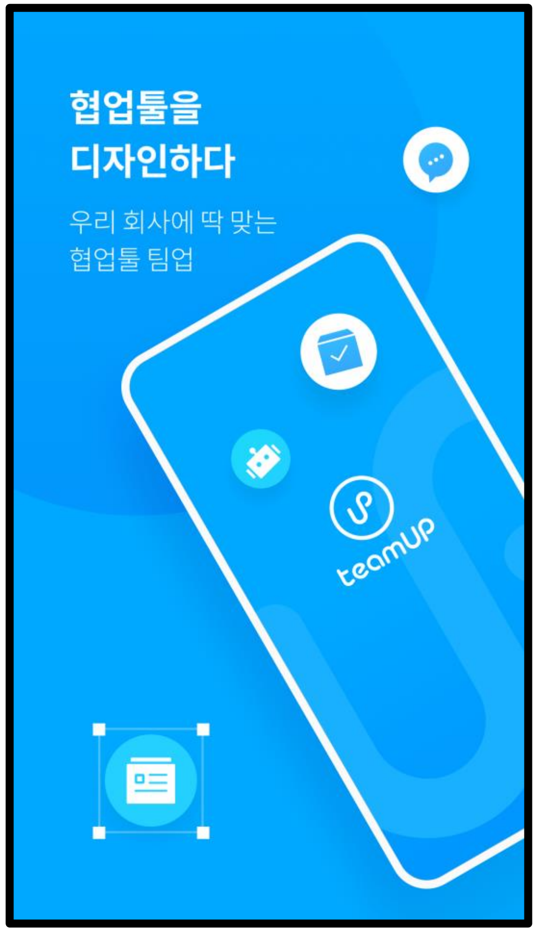
애플리케이션 분석 실습 -TeamUp

20192233 박진철

CONTENTS

1. TeamUp 소개

1. TeamUp 소개

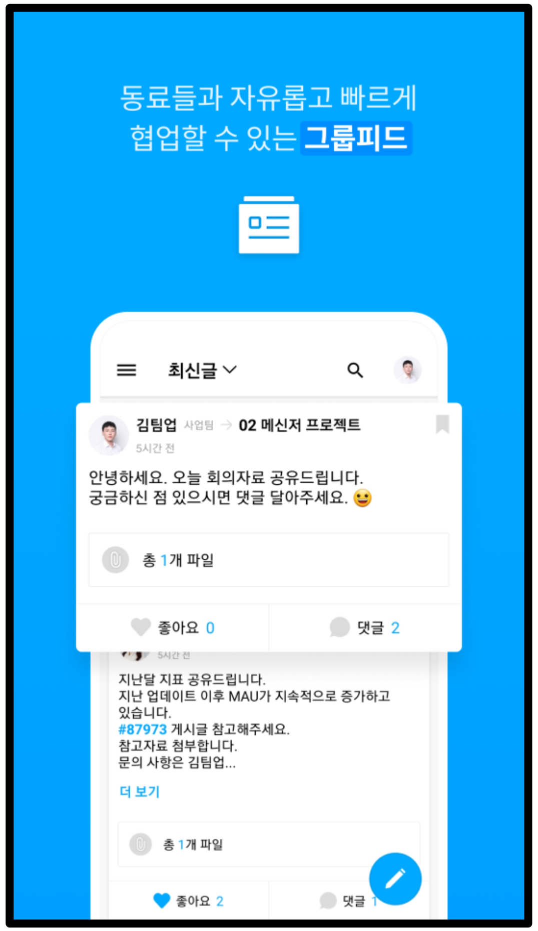


☞ 이스트소프트에서 2014년 개발

☞ 업무에서 활용가능한 기능이 들어있는 협업툴

☞ 채팅기능을 통해 팀원과 소통 가능

1. TeamUp 소개


동료들과 자유롭고 빠르게
협업할 수 있는 **그룹피드**

김팀업 사업팀 → 02 메신저 프로젝트
5시간 전

안녕하세요. 오늘 회의자료 공유드립니다.
궁금하신 점 있으시면 댓글 달아주세요. 😀

총 1개 파일

좋아요 0 | 댓글 2

5시간 전

지난달 지표 공유드립니다.
지난 업데이트 이후 MAU가 지속적으로 증가하고 있습니다.
#87973 게시글 참고해주세요.
참고자료 첨부합니다.
문의 사항은 김팀업...

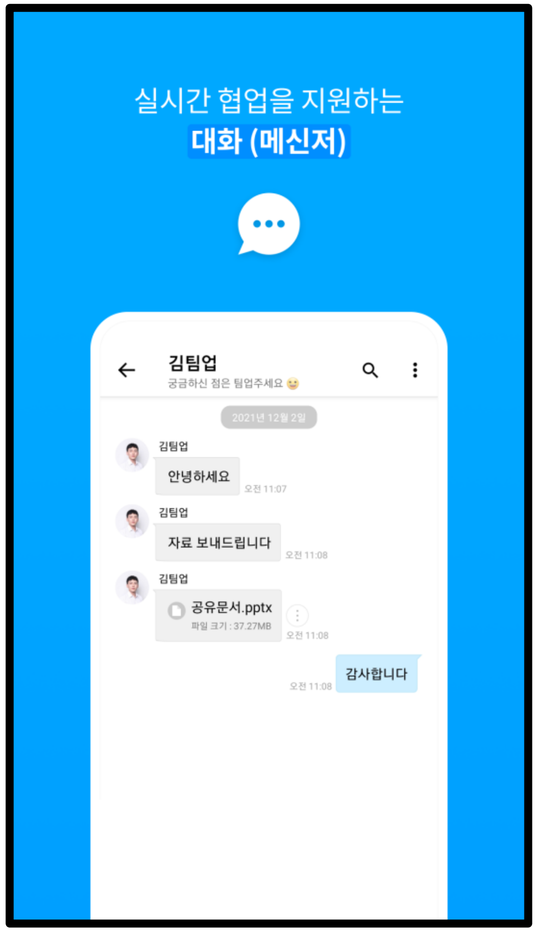
더 보기

총 1개 파일

좋아요 2 | 댓글 1

- ☞ 이스트소프트에서 2014년 개발
- ☞ 업무에서 활용가능한 기능이 들어있는 협업툴
- ☞ 채팅기능을 통해 팀원과 소통 가능

1. TeamUp 소개


실시간 협업을 지원하는
대화 (메신저)

김팀업
궁금하신 점은 팀업주세요 😀

2021년 12월 2일

김팀업
안녕하세요 오전 11:07

김팀업
자료 보내드립니다 오전 11:08

김팀업
공유문서.pptx
파일 크기 : 37.27MB 오전 11:08

오전 11:08 감사합니다

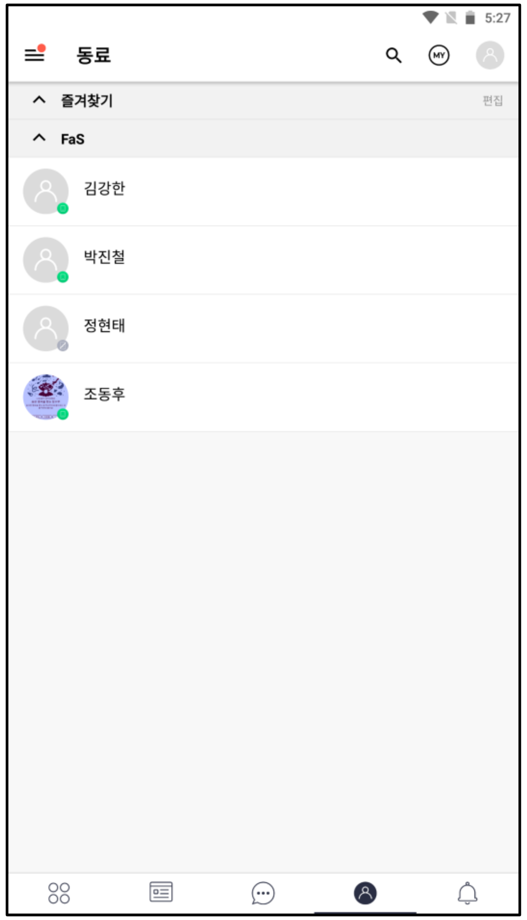
👉 이스트소프트에서 2014년 개발

👉 업무에서 활용가능한 기능이 들어있는 협업툴

👉 채팅기능을 통해 팀원과 소통 가능

2. 아티팩트 적립과정

2. 아티팩트 적립과정



5:27

동료

즐겨찾기 편집

FaS

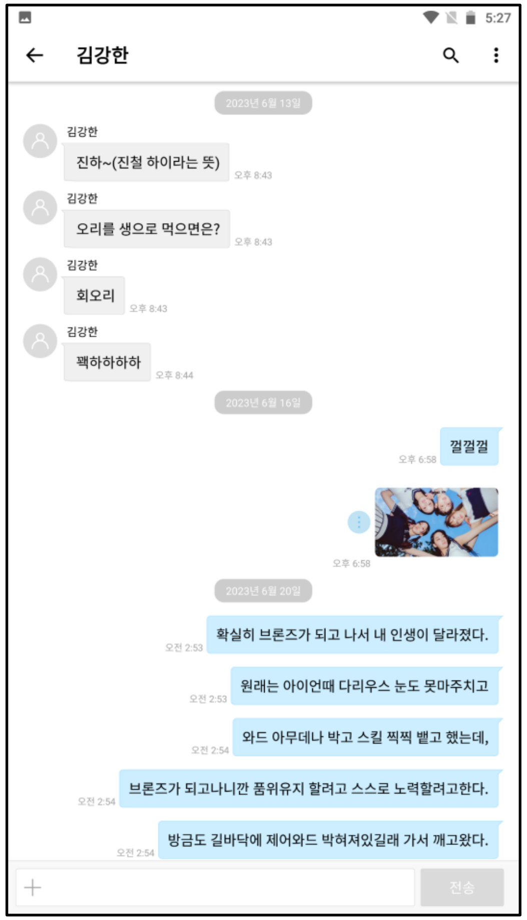
김강한

박진철

정현태

조동후

👉동아리에서 그룹 개설



5:27

김강한

2023년 6월 13일

김강한
진하~(진철 하이라는 뜻) 오후 8:43

김강한
오리를 생으로 먹으면은? 오후 8:43

김강한
회오리 오후 8:43

김강한
꽥하하하하 오후 8:44

2023년 6월 16일

오후 6:58 껄껄껄

오후 6:58

2023년 6월 20일

오전 2:53 확실히 브론즈가 되고 나서 내 인생이 달라졌다.

오전 2:53 원래는 아이언때 다리우스 눈도 못마주치고

오전 2:54 와드 아무데나 박고 스킬 찍찍 뱉고 했는데,

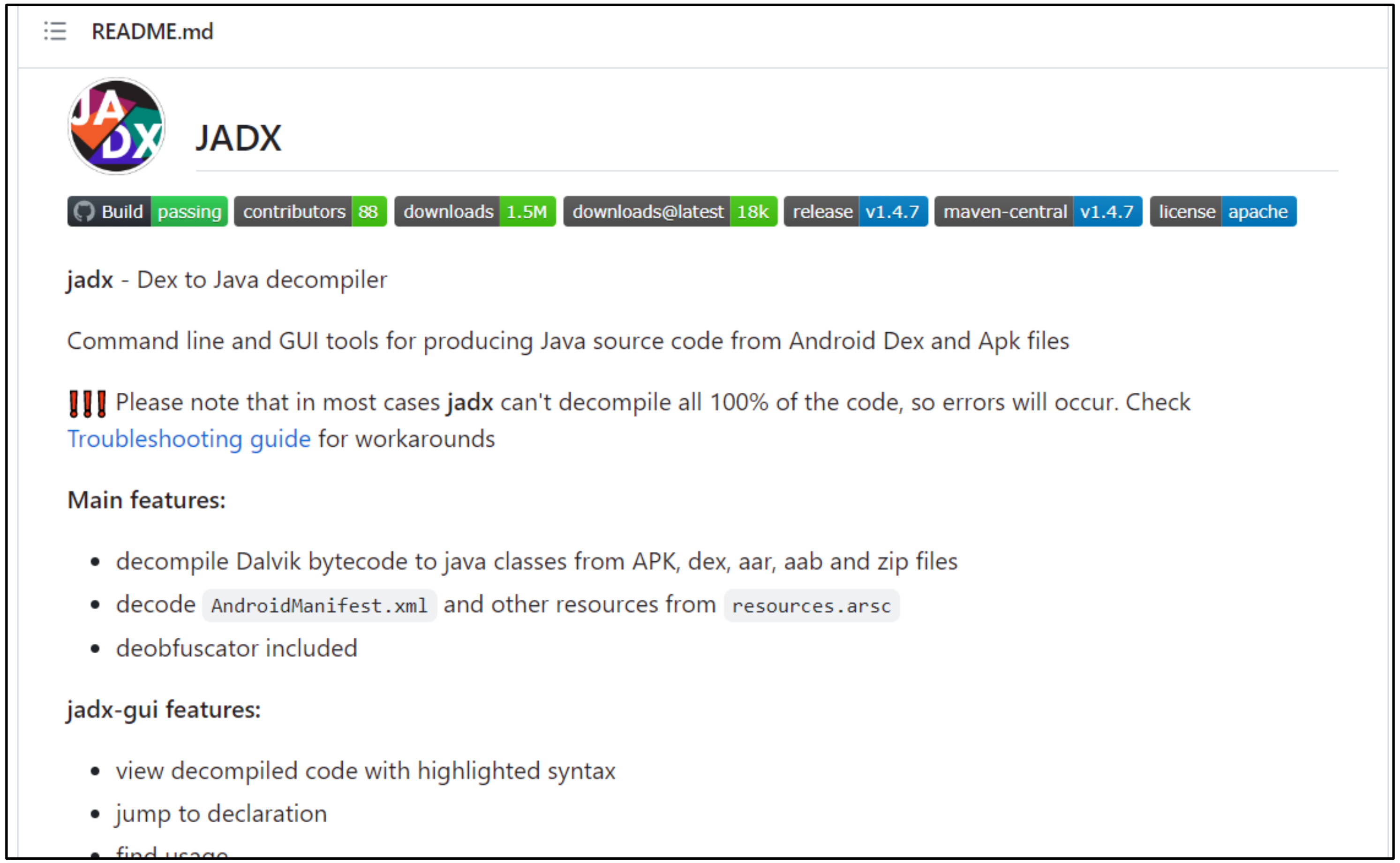
오전 2:54 브론즈가 되고나니깐 품위유지 할려고 스스로 노력할려고한다.

오전 2:54 방금도 길바닥에 제어와드 박혀져있길래 가서 깨고왔다.

전송

👉채팅을 통해 다양한 메시지 전송

2. 아티팩트 적립과정


README.md

JADX

Build passing | contributors 88 | downloads 1.5M | downloads@latest 18k | release v1.4.7 | maven-central v1.4.7 | license apache

jadx - Dex to Java decompiler

Command line and GUI tools for producing Java source code from Android Dex and Apk files

❗❗❗ Please note that in most cases **jadx** can't decompile all 100% of the code, so errors will occur. Check Troubleshooting guide for workarounds

Main features:

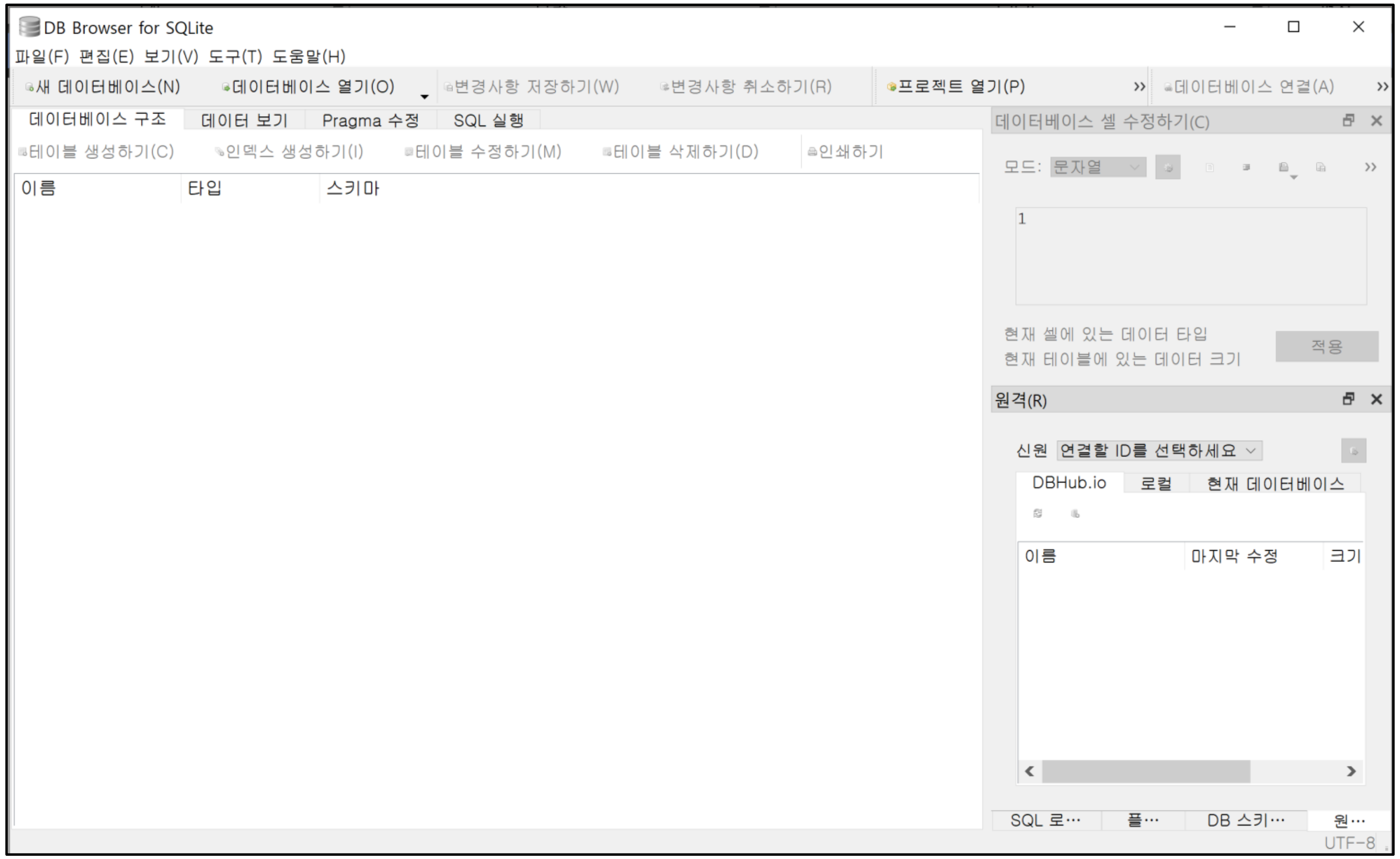
- decompile Dalvik bytecode to java classes from APK, dex, aar, aab and zip files
- decode `AndroidManifest.xml` and other resources from `resources.arsc`
- deobfuscator included

jadx-gui features:

- view decompiled code with highlighted syntax
- jump to declaration
- find usage

👉 jadx를 통해 APK파일 분석

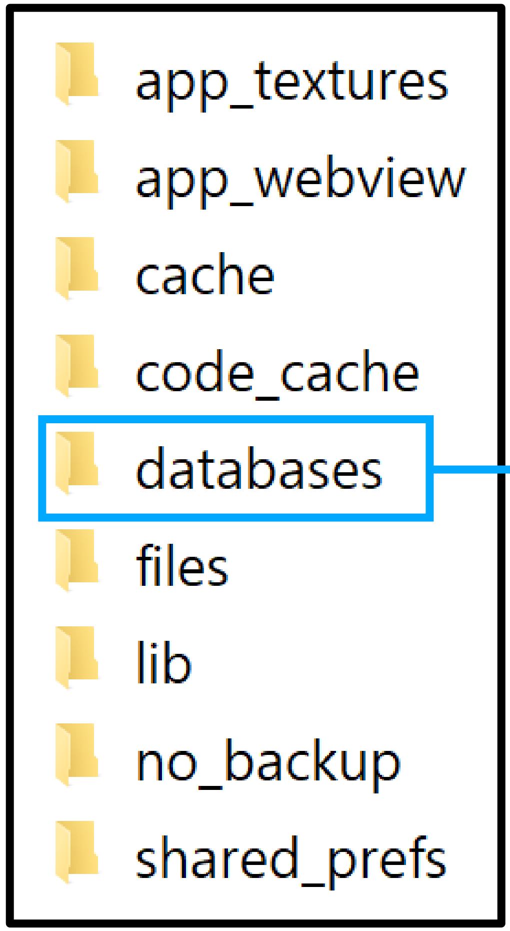
2. 아티팩트 적립과정

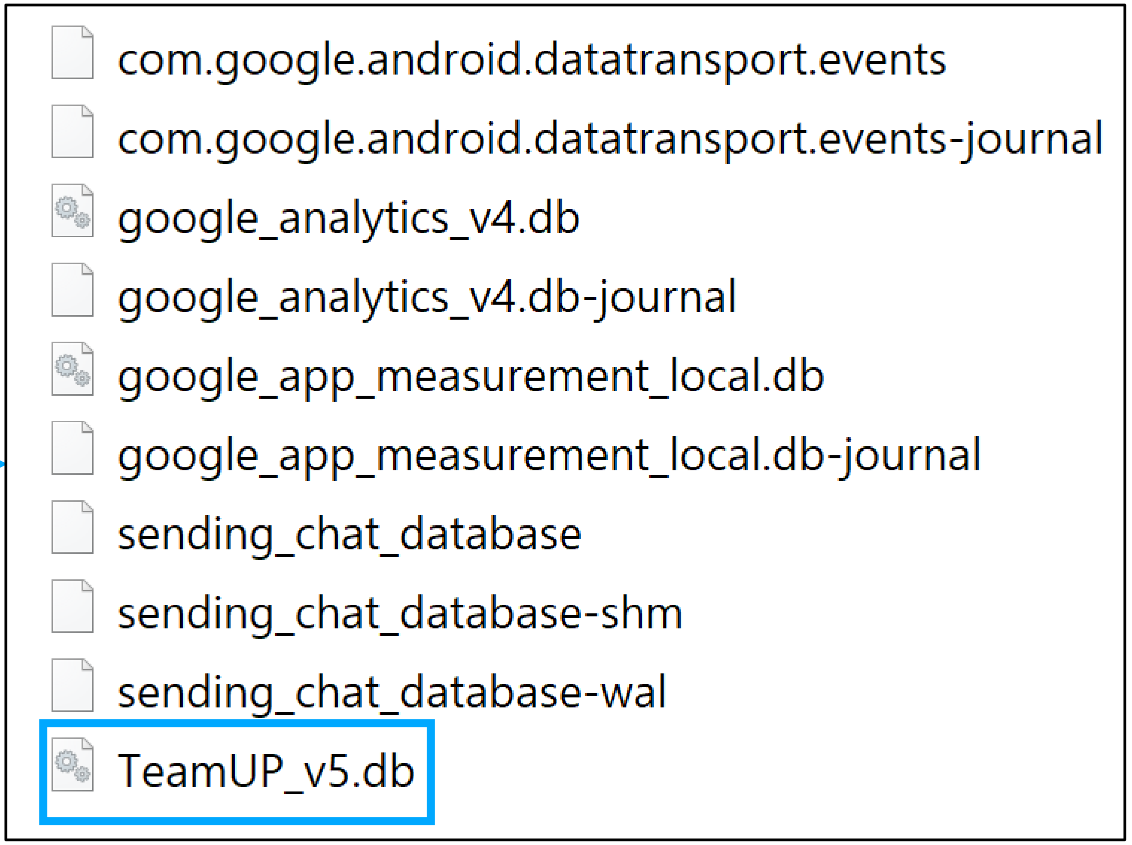


☞ DB Browser를 통해 데이터 베이스 분석

3. 데이터 파일 분석

3. 데이터 파일 분석

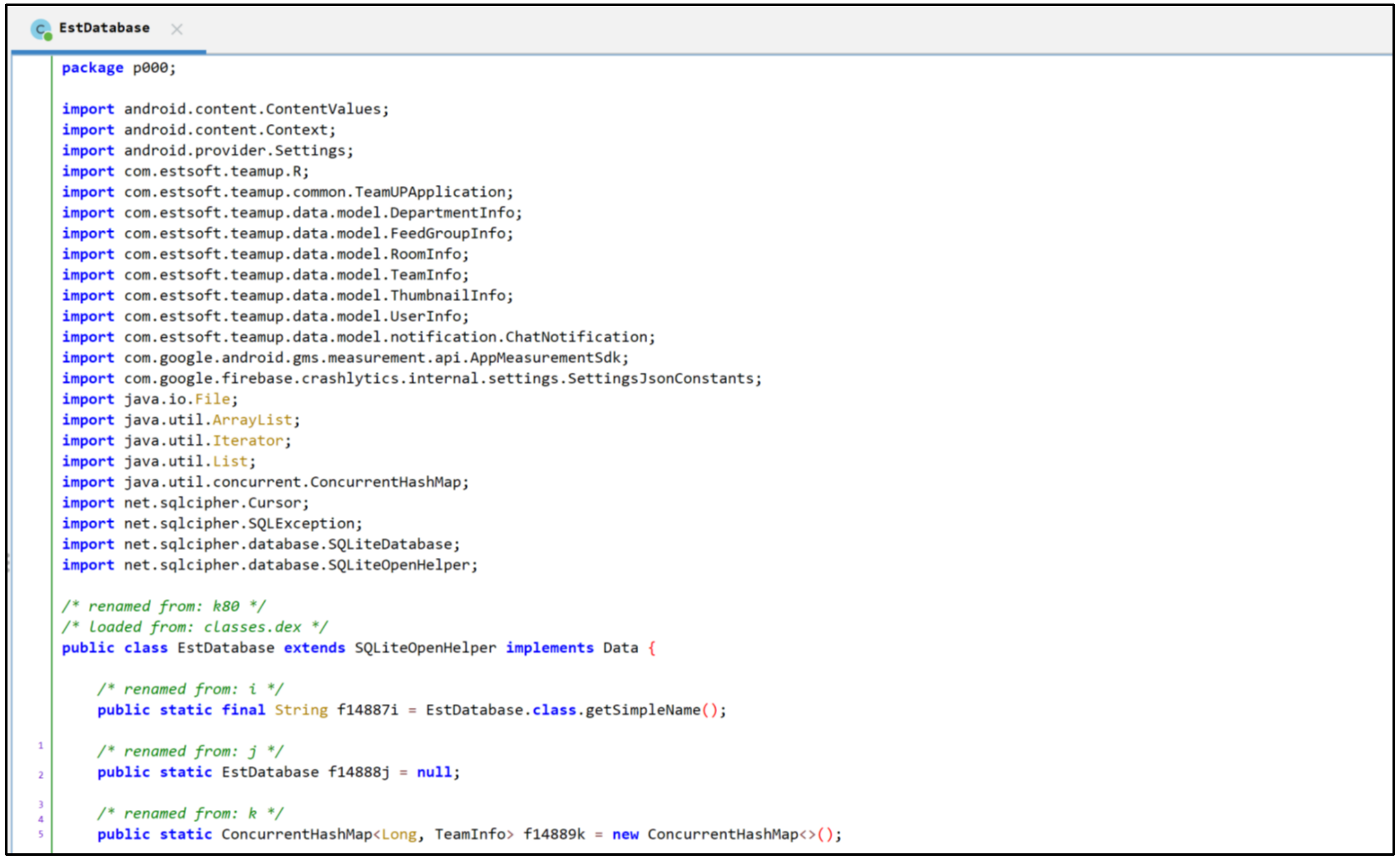

app_textures
app_webview
cache
code_cache
databases
files
lib
no_backup
shared_prefs


com.google.android.datatransport.events
com.google.android.datatransport.events-journal
google_analytics_v4.db
google_analytics_v4.db-journal
google_app_measurement_local.db
google_app_measurement_local.db-journal
sending_chat_database
sending_chat_database-shm
sending_chat_database-wal
TeamUP_v5.db

☞ databases폴더에 TeamUP_v5.db 데이터베이스 존재

4. 데이터 베이스 복호화

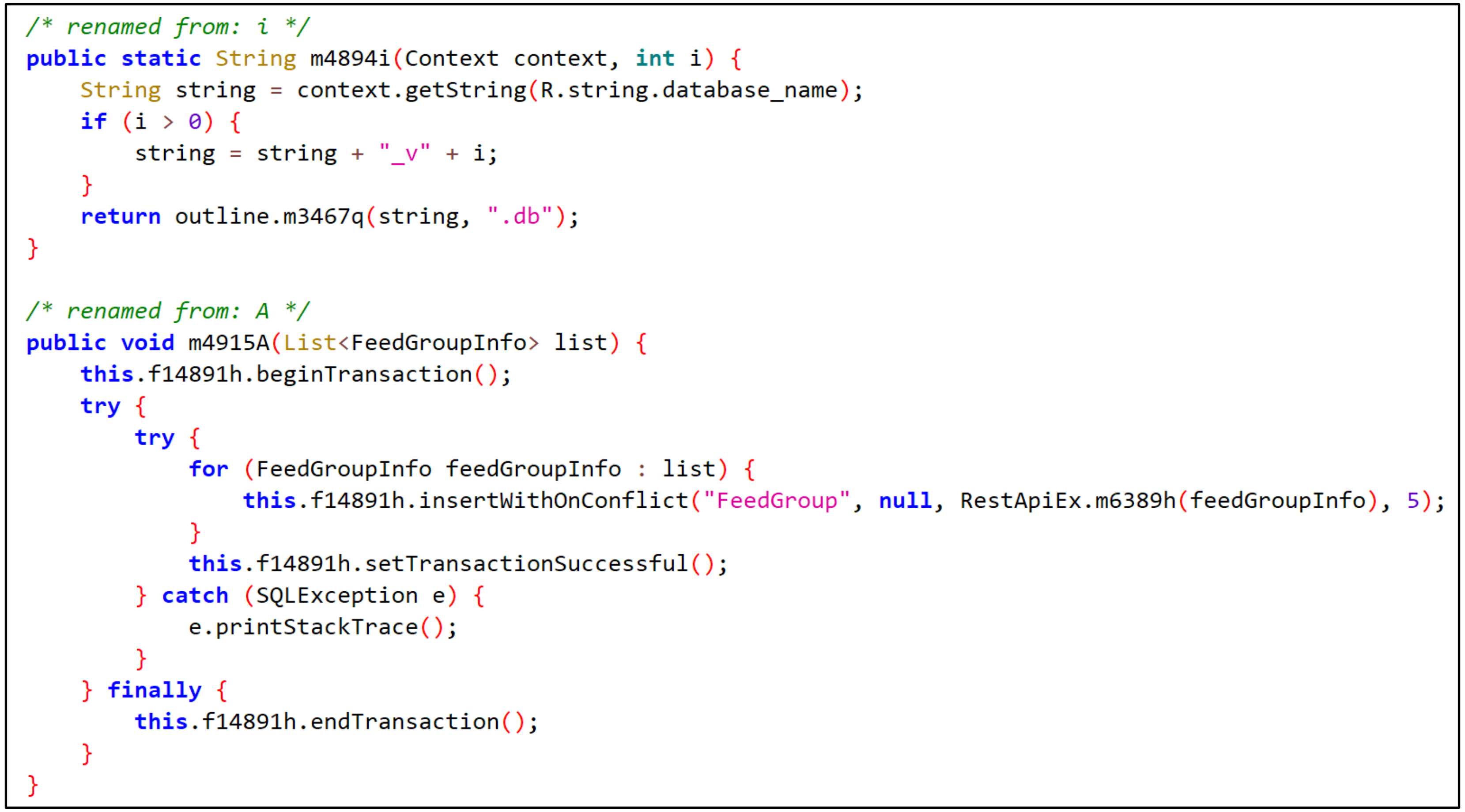
4. 데이터 베이스 복호화



```java
package p000;

import android.content.ContentValues;
import android.content.Context;
import android.provider.Settings;
import com.estsoft.teamup.R;
import com.estsoft.teamup.common.TeamUPApplication;
import com.estsoft.teamup.data.model.DepartmentInfo;
import com.estsoft.teamup.data.model.FeedGroupInfo;
import com.estsoft.teamup.data.model.RoomInfo;
import com.estsoft.teamup.data.model.TeamInfo;
import com.estsoft.teamup.data.model.ThumbnailInfo;
import com.estsoft.teamup.data.model.UserInfo;
import com.estsoft.teamup.data.model.notification.ChatNotification;
import com.google.android.gms.measurement.api.AppMeasurementSdk;
import com.google.firebase.crashlytics.internal.settings.SettingsJsonConstants;
import java.io.File;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.concurrent.ConcurrentHashMap;
import net.sqlcipher.Cursor;
import net.sqlcipher.SQLException;
import net.sqlcipher.database.SQLiteDatabase;
import net.sqlcipher.database.SQLiteOpenHelper;

/* renamed from: k80 */
/* loaded from: classes.dex */
public class EstDatabase extends SQLiteOpenHelper implements Data {

    /* renamed from: i */
    public static final String f14887i = EstDatabase.class.getSimpleName();

    /* renamed from: j */
    public static EstDatabase f14888j = null;

    /* renamed from: k */
    public static ConcurrentHashMap<Long, TeamInfo> f14889k = new ConcurrentHashMap<>();
```

☞ EstDatabase 클래스에서 TeamUP_v5.db 생성

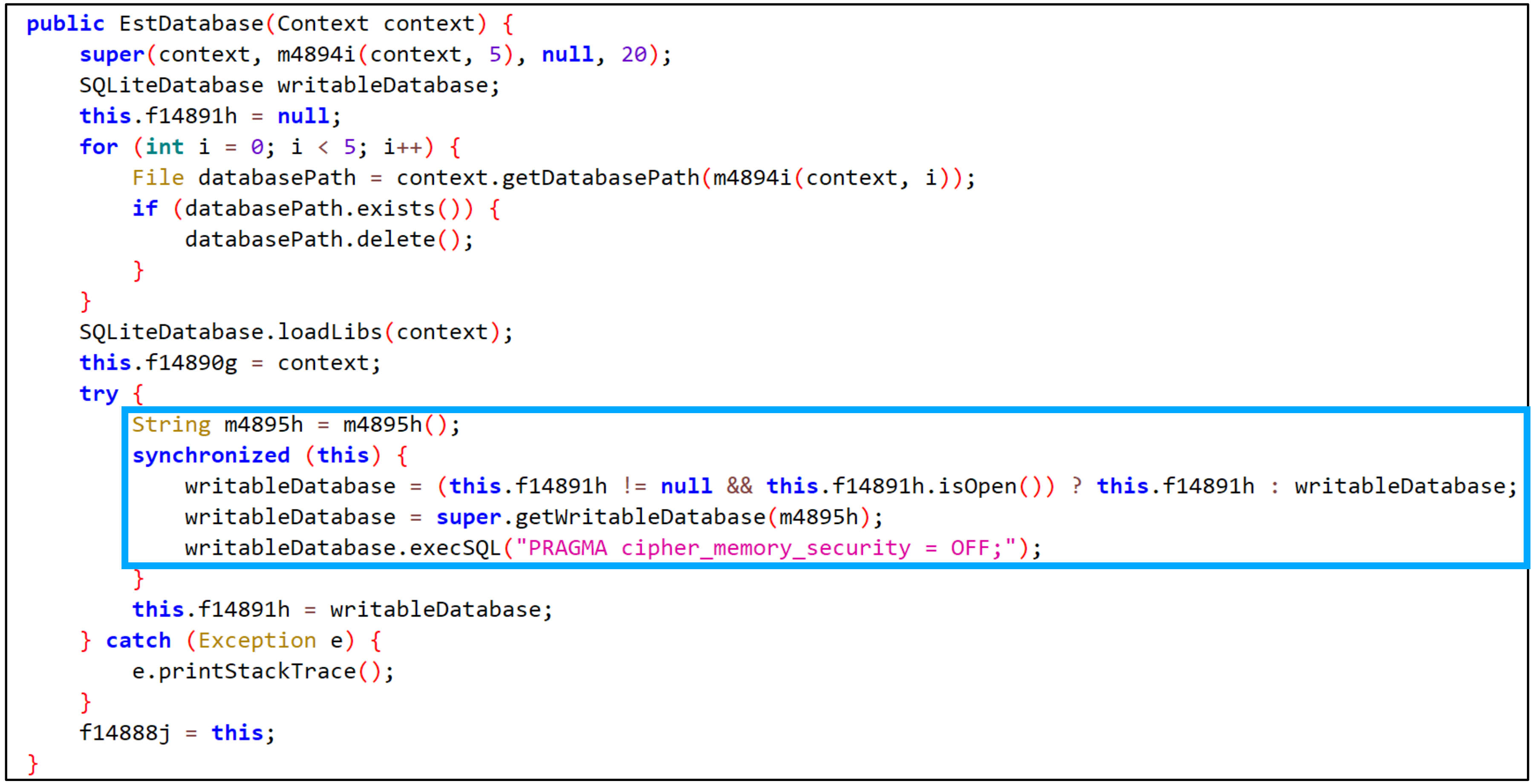
4. 데이터 베이스 복호화



```java
/* renamed from: i */
public static String m4894i(Context context, int i) {
    String string = context.getString(R.string.database_name);
    if (i > 0) {
        string = string + "_v" + i;
    }
    return outline.m3467q(string, ".db");
}

/* renamed from: A */
public void m4915A(List<FeedGroupInfo> list) {
    this.f14891h.beginTransaction();
    try {
        try {
            for (FeedGroupInfo feedGroupInfo : list) {
                this.f14891h.insertWithOnConflict("FeedGroup", null, RestApiEx.m6389h(feedGroupInfo), 5);
            }
            this.f14891h.setTransactionSuccessful();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    } finally {
        this.f14891h.endTransaction();
    }
}
```

☞ EstDatabase 클래스에서 TeamUP_v5.db 생성

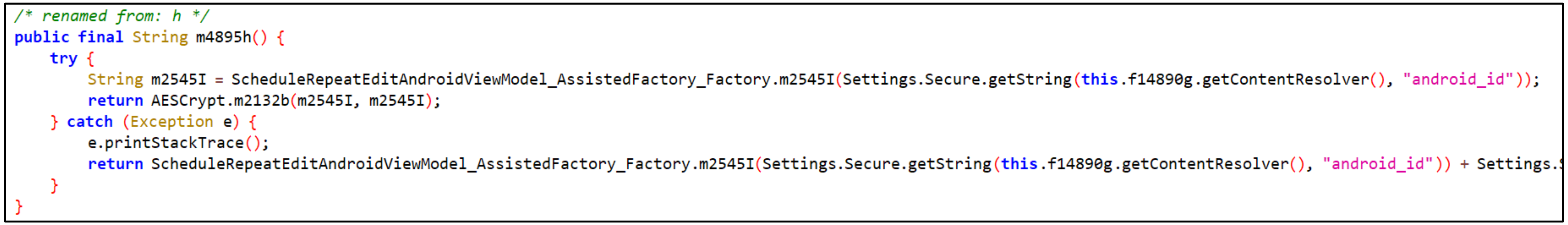
4. 데이터 베이스 복호화



```java
public EstDatabase(Context context) {
    super(context, m4894i(context, 5), null, 20);
    SQLiteDatabase writableDatabase;
    this.f14891h = null;
    for (int i = 0; i < 5; i++) {
        File databasePath = context.getDatabasePath(m4894i(context, i));
        if (databasePath.exists()) {
            databasePath.delete();
        }
    }
    SQLiteDatabase.loadLibs(context);
    this.f14890g = context;
    try {
        String m4895h = m4895h();
        synchronized (this) {
            writableDatabase = (this.f14891h != null && this.f14891h.isOpen()) ? this.f14891h : writableDatabase;
            writableDatabase = super.getWritableDatabase(m4895h);
            writableDatabase.execSQL("PRAGMA cipher_memory_security = OFF;");
        }
        this.f14891h = writableDatabase;
    } catch (Exception e) {
        e.printStackTrace();
    }
    f14888j = this;
}
```

m4895h()의 리턴 값을 키로 사용함

4. 데이터 베이스 복호화



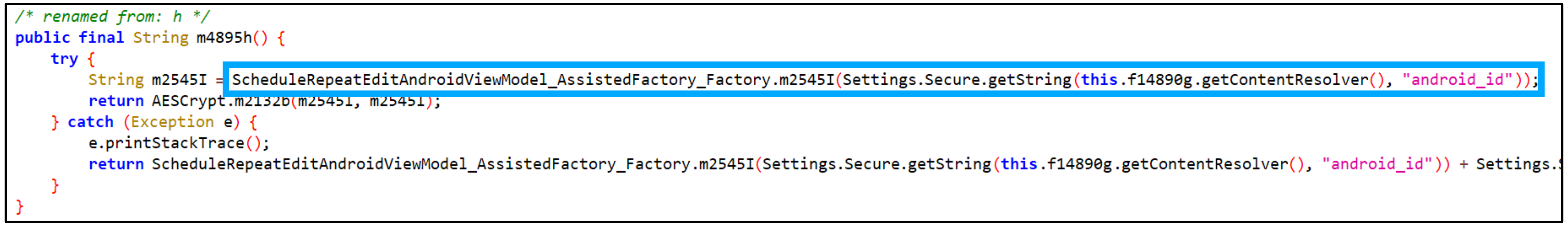
```java
/* renamed from: h */
public final String m4895h() {
    try {
        String m2545I = ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id"));
        return AESCrypt.m2132b(m2545I, m2545I);
    } catch (Exception e) {
        e.printStackTrace();
        return ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id")) + Settings.S
    }
}
```

☞ m4895h()

☞ android_id를 m2545I로 저장한 후, m2132b()에 넣은 값을 받음

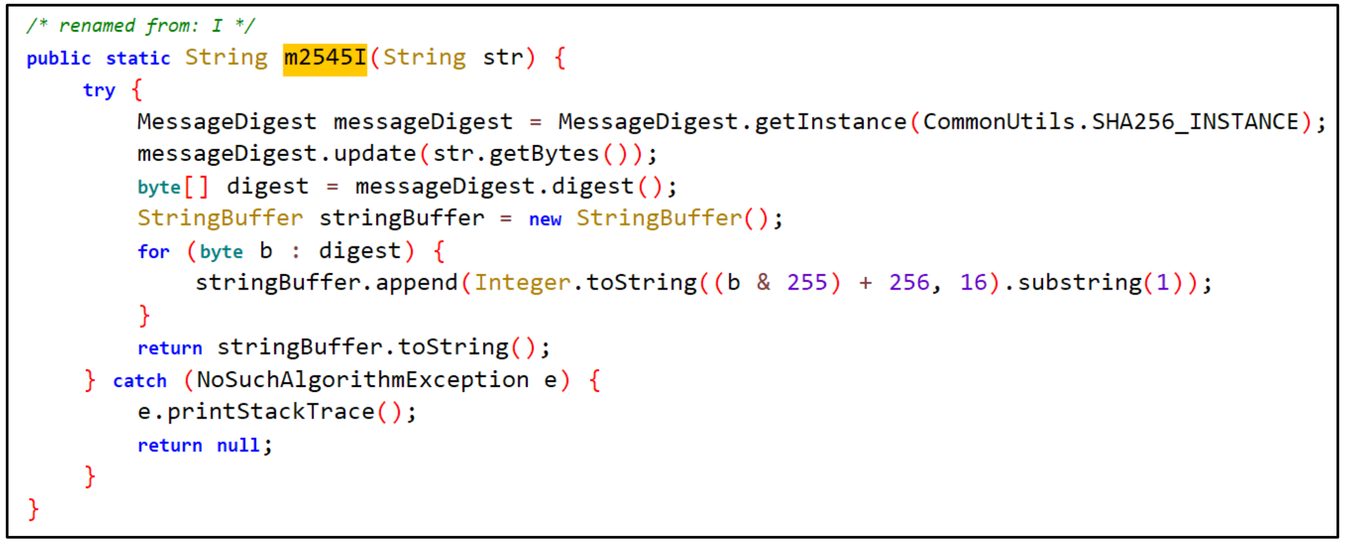
- 👍 android_id: 기기의 고유한 식별자
- 👍 기기마다 다른 값이 나옴 →디바이스를 구별하는 용도로 사용
- 👎 아직 찾지 못함…

4. 데이터 베이스 복호화



```java
/* renamed from: h */
public final String m4895h() {
    try {
        String m2545I = ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id"));
        return AESCrypt.m2132b(m2545I, m2545I);
    } catch (Exception e) {
        e.printStackTrace();
        return ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id")) + Settings.
    }
}
```

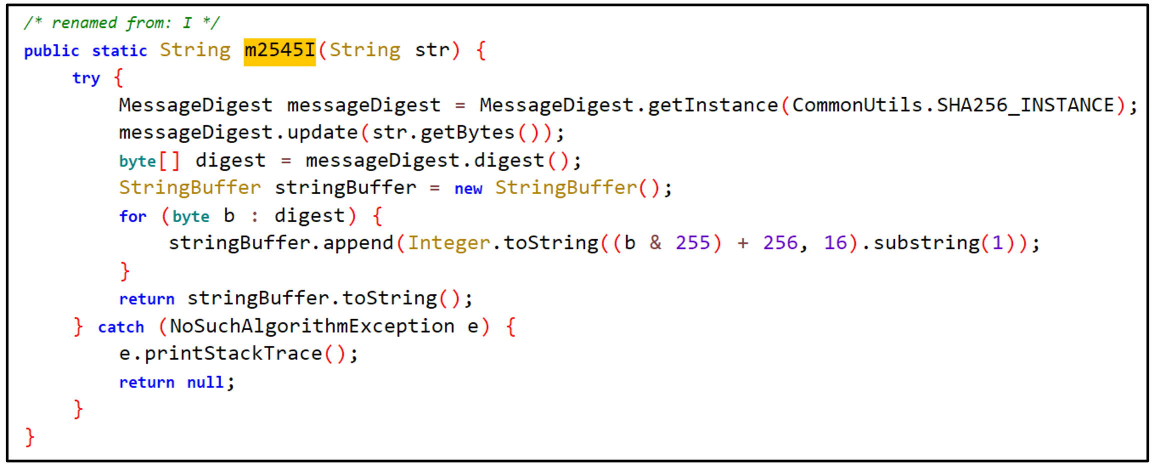
☞ m4895h()



```java
/* renamed from: I */
public static String m2545I(String str) {
    try {
        MessageDigest messageDigest = MessageDigest.getInstance(CommonUtils.SHA256_INSTANCE);
        messageDigest.update(str.getBytes());
        byte[] digest = messageDigest.digest();
        StringBuffer stringBuffer = new StringBuffer();
        for (byte b : digest) {
            stringBuffer.append(Integer.toString((b & 255) + 256, 16).substring(1));
        }
        return stringBuffer.toString();
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
        return null;
    }
}
```

☞ m2545I()

4. 데이터 베이스 복호화

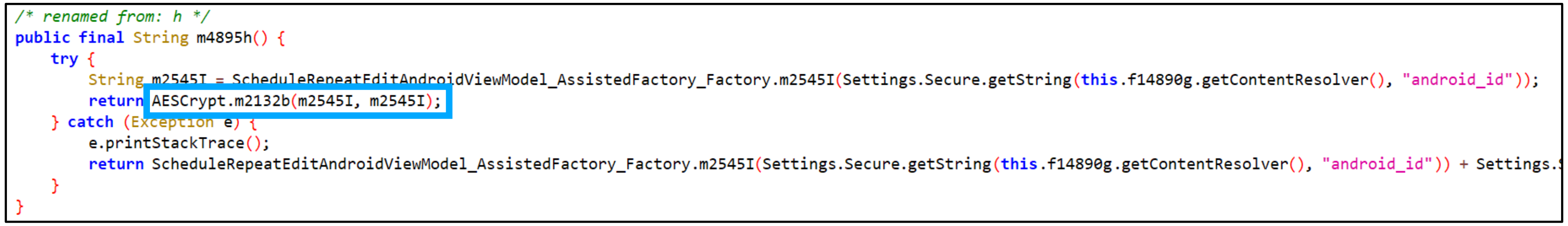


```java
/* renamed from: I */
public static String m2545I(String str) {
    try {
        MessageDigest messageDigest = MessageDigest.getInstance(CommonUtils.SHA256_INSTANCE);
        messageDigest.update(str.getBytes());
        byte[] digest = messageDigest.digest();
        StringBuffer stringBuffer = new StringBuffer();
        for (byte b : digest) {
            stringBuffer.append(Integer.toString((b & 255) + 256, 16).substring(1));
        }
        return stringBuffer.toString();
    } catch (NoSuchAlgorithmException e) {
        e.printStackTrace();
        return null;
    }
}
```

☞ m2545i(android_id) sudo 코드

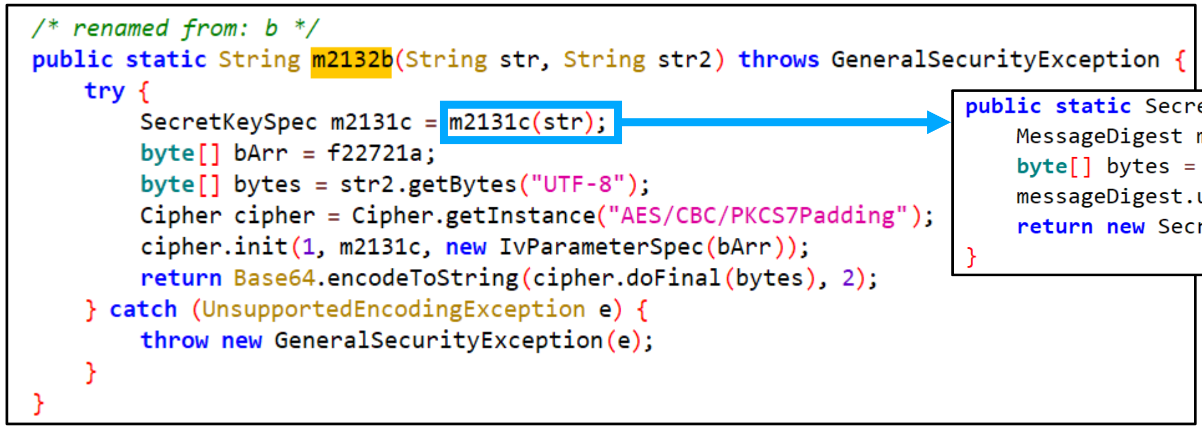
```
InPut: android_id
OutPut: str_buffer

digest <- SHA_256(bytes(android_id))
for b in digest:
        dig <- hex((b&255)+256)
        str_buffer+=dig[1:]
```

4. 데이터 베이스 복호화

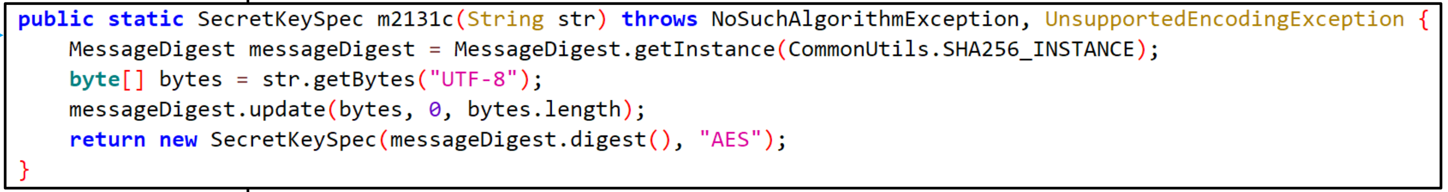


```java
/* renamed from: h */
public final String m4895h() {
    try {
        String m2545I = ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id"));
        return AESCrypt.m2132b(m2545I, m2545I);
    } catch (Exception e) {
        e.printStackTrace();
        return ScheduleRepeatEditAndroidViewModel_AssistedFactory_Factory.m2545I(Settings.Secure.getString(this.f14890g.getContentResolver(), "android_id")) + Settings.
    }
}
```

☞ m4895h()



```java
/* renamed from: b */
public static String m2132b(String str, String str2) throws GeneralSecurityException {
    try {
        SecretKeySpec m2131c = m2131c(str);
        byte[] bArr = f22721a;
        byte[] bytes = str2.getBytes("UTF-8");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS7Padding");
        cipher.init(1, m2131c, new IvParameterSpec(bArr));
        return Base64.encodeToString(cipher.doFinal(bytes), 2);
    } catch (UnsupportedEncodingException e) {
        throw new GeneralSecurityException(e);
    }
}
```

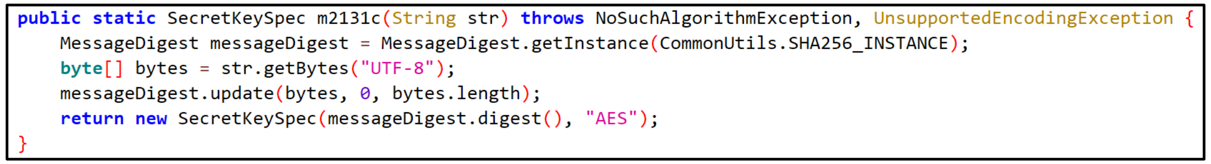


```java
public static SecretKeySpec m2131c(String str) throws NoSuchAlgorithmException, UnsupportedEncodingException {
    MessageDigest messageDigest = MessageDigest.getInstance(CommonUtils.SHA256_INSTANCE);
    byte[] bytes = str.getBytes("UTF-8");
    messageDigest.update(bytes, 0, bytes.length);
    return new SecretKeySpec(messageDigest.digest(), "AES");
}
```

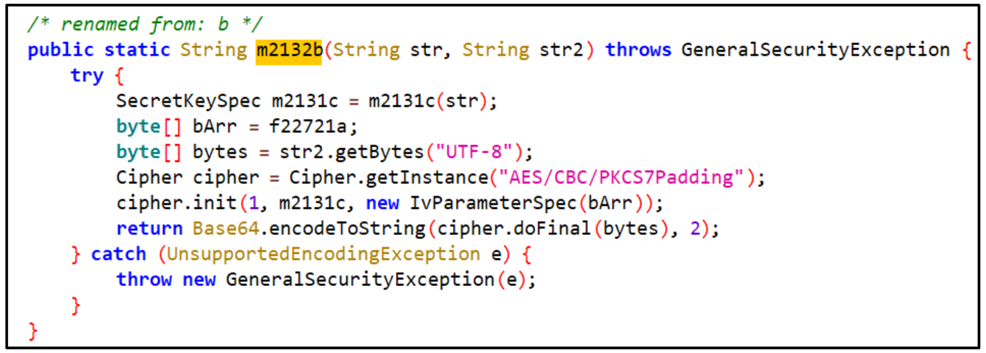
☞ m2131c()

☞ m2132b()

4. 데이터 베이스 복호화



```java
public static SecretKeySpec m2131c(String str) throws NoSuchAlgorithmException, UnsupportedEncodingException {
    MessageDigest messageDigest = MessageDigest.getInstance(CommonUtils.SHA256_INSTANCE);
    byte[] bytes = str.getBytes("UTF-8");
    messageDigest.update(bytes, 0, bytes.length);
    return new SecretKeySpec(messageDigest.digest(), "AES");
}
```



```java
/* renamed from: b */
public static String m2132b(String str, String str2) throws GeneralSecurityException {
    try {
        SecretKeySpec m2131c = m2131c(str);
        byte[] bArr = f22721a;
        byte[] bytes = str2.getBytes("UTF-8");
        Cipher cipher = Cipher.getInstance("AES/CBC/PKCS7Padding");
        cipher.init(1, m2131c, new IvParameterSpec(bArr));
        return Base64.encodeToString(cipher.doFinal(bytes), 2);
    } catch (UnsupportedEncodingException e) {
        throw new GeneralSecurityException(e);
    }
}
```

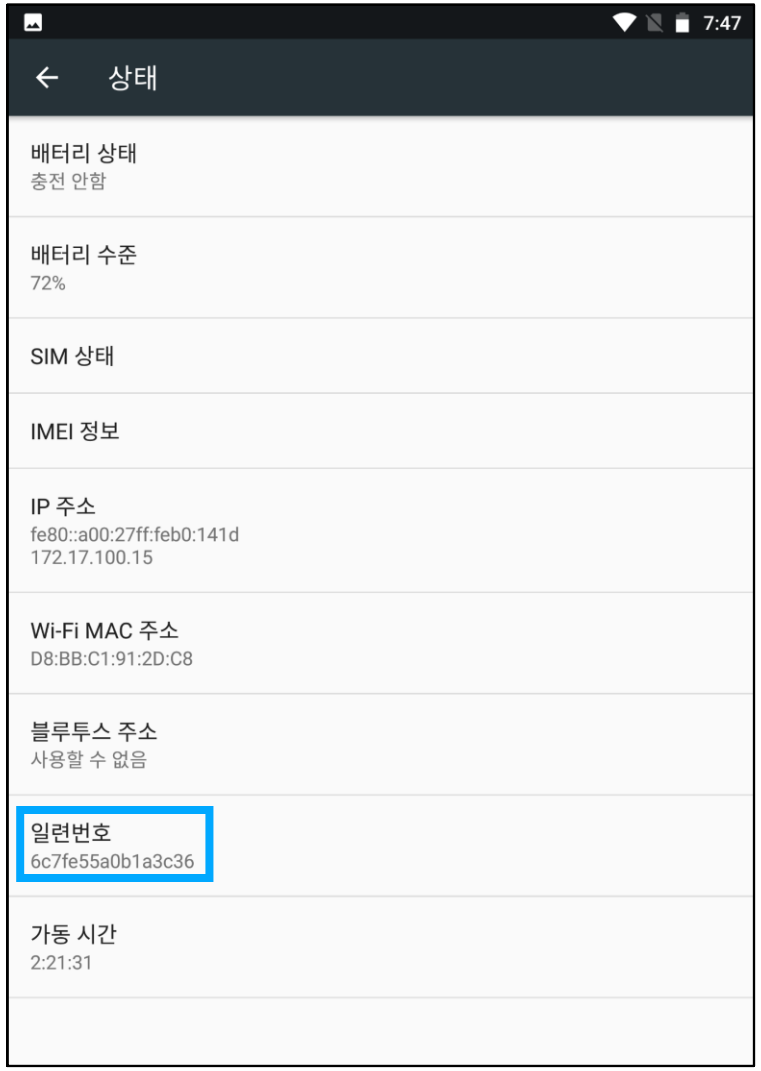
☞ m2131c(str) sudo 코드

```
InPut: str
OutPut: msg_digest

msg_digest <- SHA_256(UTF-8_Encode(str))
```

☞ m2132b(m2545l) sudo 코드

```
InPut: and_id(=m2545I)
OutPut: sql_key

key <- m2131c(and_id)
IV <- 0
bytes <- UTF-8_Encode(and_id)
enc_byte <- AES256_Encrypt(bytes,key,IV)
Sql_key <- Base64_Encode(enc_byte)
```

4. 데이터 베이스 복호화



7:47

상태

배터리 상태
충전 안함

배터리 수준
72%

SIM 상태

IMEI 정보

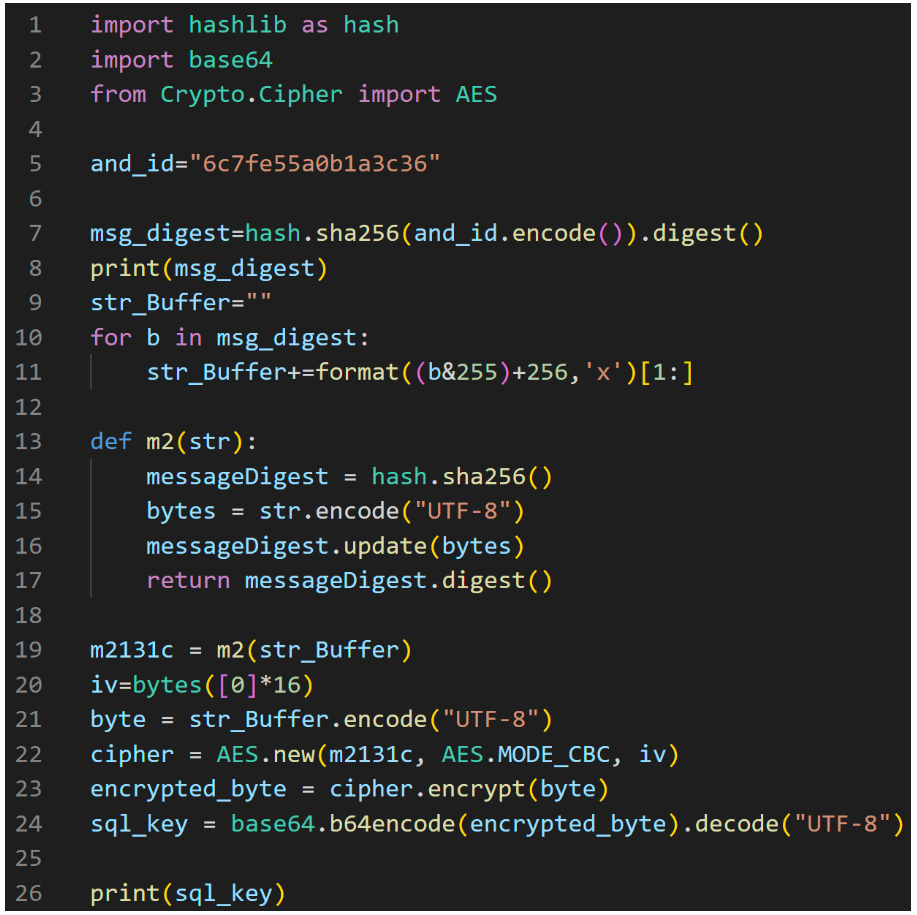
IP 주소
fe80::a00:27ff:feb0:141d
172.17.100.15

Wi-Fi MAC 주소
D8:BB:C1:91:2D:C8

블루투스 주소
사용할 수 없음

일련번호
6c7fe55a0b1a3c36

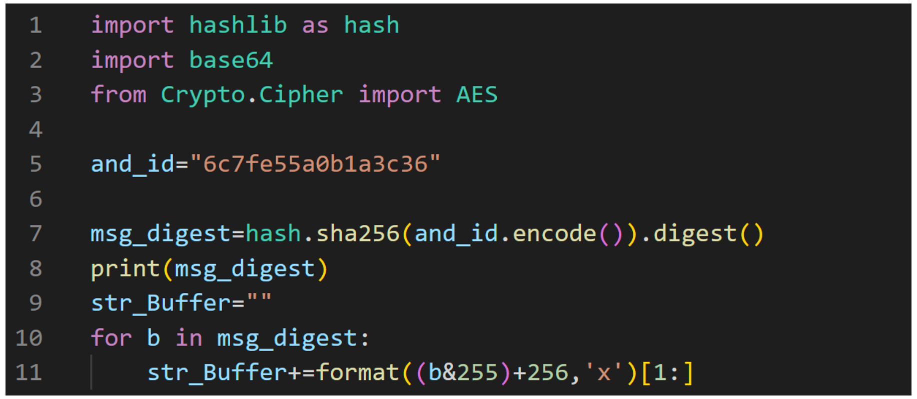
가동 시간
2:21:31



```python
import hashlib as hash
import base64
from Crypto.Cipher import AES

and_id="6c7fe55a0b1a3c36"

msg_digest=hash.sha256(and_id.encode()).digest()
print(msg_digest)
str_Buffer=""
for b in msg_digest:
    str_Buffer+=format((b&255)+256,'x')[1:]

def m2(str):
    messageDigest = hash.sha256()
    bytes = str.encode("UTF-8")
    messageDigest.update(bytes)
    return messageDigest.digest()

m2131c = m2(str_Buffer)
iv=bytes([0]*16)
byte = str_Buffer.encode("UTF-8")
cipher = AES.new(m2131c, AES.MODE_CBC, iv)
encrypted_byte = cipher.encrypt(byte)
sql_key = base64.b64encode(encrypted_byte).decode("UTF-8")

print(sql_key)
```

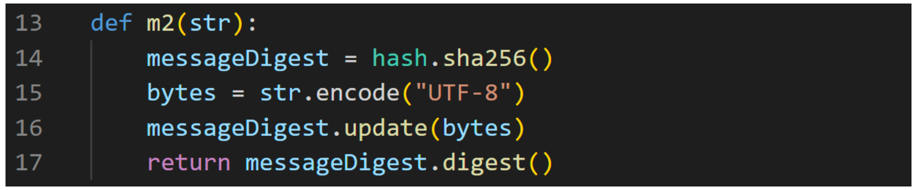
4. 데이터 베이스 복호화

```
InPut: android_id
OutPut: str_buffer

digest <- SHA_256(bytes(android_id))
for b in digest:
        dig <- hex((b&255)+256)
        str_buffer+=dig[1:]
```



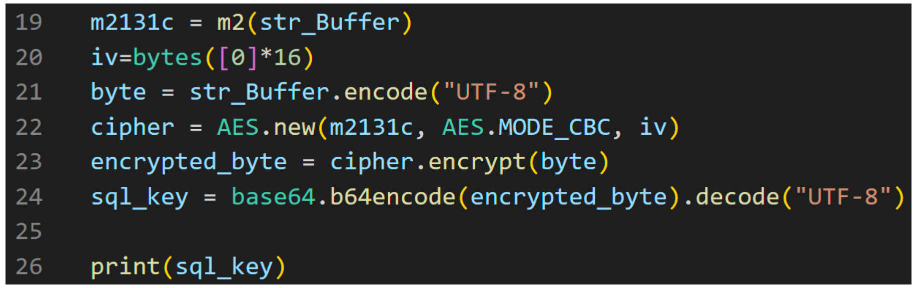
```python
import hashlib as hash
import base64
from Crypto.Cipher import AES

and_id="6c7fe55a0b1a3c36"

msg_digest=hash.sha256(and_id.encode()).digest()
print(msg_digest)
str_Buffer=""
for b in msg_digest:
    str_Buffer+=format((b&255)+256,'x')[1:]
```

```
InPut: str
OutPut: msg_digest

msg_digest <- SHA_256(UTF-8_Encode(str))
```



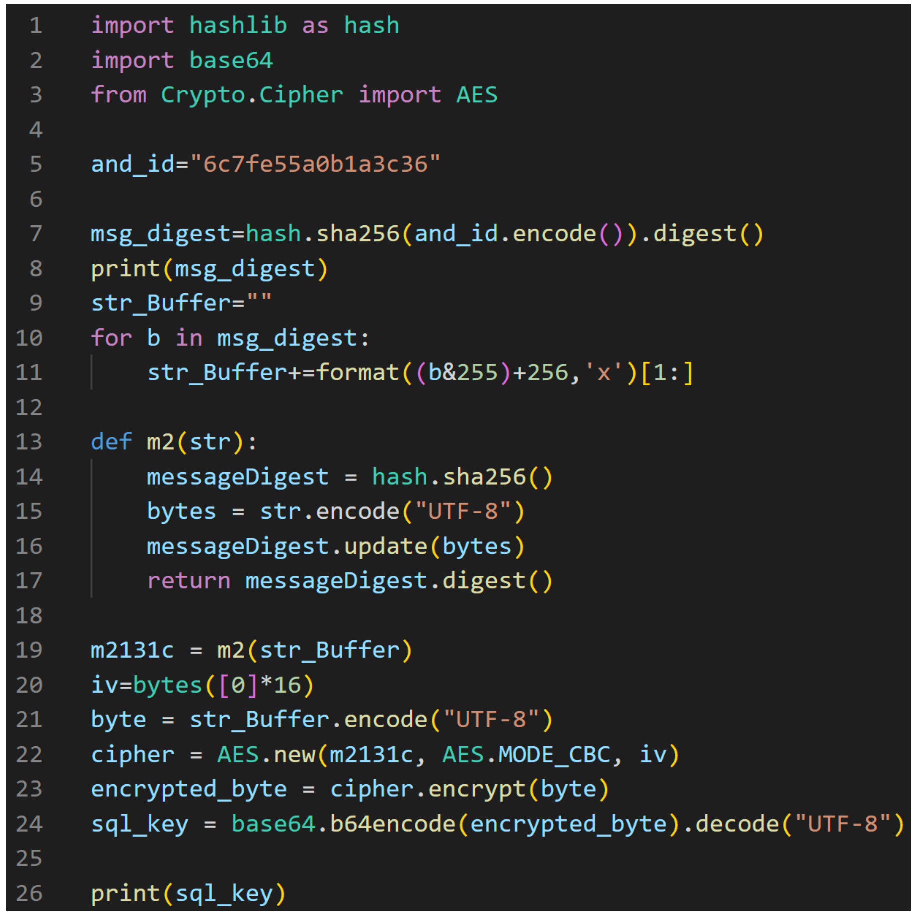
```python
def m2(str):
    messageDigest = hash.sha256()
    bytes = str.encode("UTF-8")
    messageDigest.update(bytes)
    return messageDigest.digest()
```

```
InPut: and_id(=m2545I)
OutPut: sql_key

key <- m2131c(and_id)
IV <- 0
bytes <- UTF-8_Encode(and_id)
enc_byte <- AES256_Encrypt(bytes,key,IV)
Sql_key <- Base64_Encode(enc_byte)
```

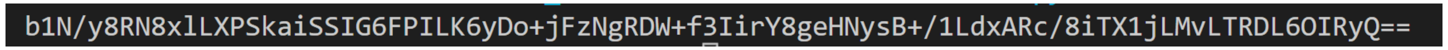


```python
m2131c = m2(str_Buffer)
iv=bytes([0]*16)
byte = str_Buffer.encode("UTF-8")
cipher = AES.new(m2131c, AES.MODE_CBC, iv)
encrypted_byte = cipher.encrypt(byte)
sql_key = base64.b64encode(encrypted_byte).decode("UTF-8")

print(sql_key)
```

4. 데이터 베이스 복호화



```python
import hashlib as hash
import base64
from Crypto.Cipher import AES

and_id="6c7fe55a0b1a3c36"

msg_digest=hash.sha256(and_id.encode()).digest()
print(msg_digest)
str_Buffer=""
for b in msg_digest:
    str_Buffer+=format((b&255)+256,'x')[1:]

def m2(str):
    messageDigest = hash.sha256()
    bytes = str.encode("UTF-8")
    messageDigest.update(bytes)
    return messageDigest.digest()

m2131c = m2(str_Buffer)
iv=bytes([0]*16)
byte = str_Buffer.encode("UTF-8")
cipher = AES.new(m2131c, AES.MODE_CBC, iv)
encrypted_byte = cipher.encrypt(byte)
sql_key = base64.b64encode(encrypted_byte).decode("UTF-8")

print(sql_key)
```



```
b1N/y8RN8xlLXPSkaiSSIG6FPILK6yDo+jFzNgRDW+f3IirY8geHNysB+/1LdxARc/8iTX1jLMvLTRDL6OIRyQ==
```

☞ 키를 만들었으나 복호화가 되지 않음

5. 해결하지 못한 부분

5. 해결하지 못한 부분

☞ android_id는 android 8.0이후는 앱마다 다른 값을 받음
→기기에 들어 있은 일련번호는 android_id가 아님

👎 TeamUp의 android_id를 얻는 방법을 찾지 못함
→android_id를 얻는 방법 찾기

☞ 만들어 놓은 파이썬이 올바른 코드인지 확인이 어려움

☞ 찾은 데이터 베이스 키가 맞는 값인지 알 수 없음

☞ 루팅폰이 아닌 녹스를 통해 데이터를 추출함
→픽셀 폰 루팅하는 법 공부

THANKS TO WATCHING

20192233 박진철